Hola Mundo Web con PHP y Facileal

Conceptos Básicos.

¿ Qué es PHP ?

PHP (acronimo de "PHP: Hypertext Preprocessor") es un lenguaje "OpenSource" interpretado de alto nivel embebido en páginas HTML y ejecutado en el servidor. Ejemplo:

```
<html>
   <head>
      <title>Example</title>
   </head>
   <body>
      <?php
               echo "Hola Mundo Web con PHP";
      ?>
   </body>
</html>
```

Lo que distingue a PHP de la tecnología Javascript, la cual se ejecuta en la máquina cliente, es que el código PHP es ejecutado en el servidor. El cliente solamente recibirá el resultado de la ejecución en el servidor, sin ninguna posibilidad de determinar que código ha producido el resultado recibido. El servidor web puede ser incluso configurado para que procese todos los ficheros HTML con PHP.

¿ Qué se puede hacer con PHP ?

Existen tres campos en los que scripts escritos en PHP son usados.

Scripts en la parte del servidor. Se necesitan tres cosas para que esto funcione. El parseador PHP (CGI ó módulo), un servidor web y un navegador. Se necesita correr el servidor web con PHP instalado. El resultado del programa PHP se puede obtener a través del navegador, conectando con el servidor web.

Scripts en linea de comandos. Puedes crear un script PHP y correrlo sin ningún servidor web ó navegador. Solamente necesitas el parseador PHP para usarlo de esta manera. Este tipo de uso es ideal para scripts ejecutados regularmente desde cron (en *nix ó Linux) ó el Planificador de tareas (en Windows).

Escribir aplicaciones gráficas clientes. PHP no es probablemente el mejor lenguaje para escribir aplicaciones gráficas, pero si sabes bien PHP, y te gustaría utilizar algunas

características avanzadas en programas clientes, puedes utilizar PHP-GTK para escribir dichos programas. Es también posible escribir aplicaciones independientes de una plataforma. PHP-GTK es una extensión de PHP, no disponible en la distribución principal. Si te interesa PHP-GTK, puedes visitar las páginas web del proyecto.

PHP puede ser utilizado en Linux, muchas variantes Unix (incluido HP-UX, Solaris y OpenBSD), Microsoft Windows, Mac OS X, RISC OS y probablemente alguno más. PHP soporta la mayoría de servidores web, incluyendo Apache, Microsoft Internet Information Server, Personal Web Server, Netscape, iPlanet, Oreilly Website Pro server, Caudium, Xitami, OmniHTTPd y muchos otros. PHP tiene módulos disponibles para la mayoría de los servidores, para aquellos otros que soporten el estándar CGI, PHP puede usarse como procesador CGI.

Entre las habilidades de PHP se incluyen, creación de imágenes, ficheros PDF y películas Flash (usando libswf y Ming). Tambien puedes presentar otros resultados, como XHTML y ficheros XML. PHP puede auto generar estos ficheros y grabarlos en el sistema de ficheros en vez de presentarlos en la pantalla.

Quizás la característica más potente y destacable de PHP es su soporte para una gran cantidad de bases de datos. Las siguientes bases de datos están soportadas actualmente:

Adabas D, dBase, Empress, FilePro, Hyperwave, IBM DB2, Informix, FrontBase, mSQL, Direct MS-SQL, MySQL, ODBC, Oracle (OCI7 and OCI8), Ovrimos, PostgreSQL, Solid, Sybase, Velocis, Unix dbm.

PHP también tiene soporte para comunicarse con otros servicios usando protocolos tales como LDAP, IMAP, SNMP, NNTP, POP3, HTTP, COM (en Windows) y muchos otros. También se pueden crear raw sockets.

PHP tiene unas características muy útiles para el proceso de texto, desde expresiones regulares POSIX Extended ó Perl hasta parseador de documentos XML, puedes utilizar la extensión XSLT para transformar documentos XML, en el campo del comercio electrónico, encontraras muy útiles las funciones Cybercash, CyberMUT, VeriSign Payflow Pro y CCVS, las funciones del motor de búsquedas mnoGoSearch, funciones para pasarelas de IRC, utilidades de compresión gzip, bz2, conversión de calendarios, traducción y algunas cuantas opciones mas.

¿ Que necesito ?

Si lo que quieres es hacer páginas web interactivas, necesitaras un servidor de paginas, el parseador de PHP y también puedes utilizar un manejador de bases de datos.

Puedes visitar los sitios de:
PHP: http://www.php.net

Apache: http://www.apache.org
MySQL: http://www.mysql.com

Si trabajas en un sistema operativo Windows, en la siguiente página podrás encontrar un pequeño tutorial de como instalar estas tres herramientas.
http://consol.interlegit.com.mx

También puedes buscar otras herramientas mas, algunas de ellas pueden instalarte PHP, Apache y MySQL en pocos pasos, o quizás utilizar otros servidores web y manejadores de bases de datos.

Sintaxis.

Saliendo de HTML

PHP utiliza las etiquetas epeciles <?php y ?> para indicar el comienzo y final del bloque de instrucciones PHP, el siguiente ejemplo es muy claro.

```
<html>
    <head>
        <title>PHP</title>
    </head>
    <body>
        Esto es HTML<br><br>
        <?php
            echo “Esto es PHP<br><br>”;
        ?>
            <div style=”color:red”><?php echo “más datos aquí</div>”; ?>
    </body>
    </html>
```

Separación de instrucciones

La separación de instrucciones se hace de la misma manera que en C o Perl, terminando cada declaración con un punto y coma.

La etiqueta de fin de bloque (?>) implica el fin de la declaración, por lo tanto lo siguiente es equivalente:

```
<?php echo "Esto es una prueba"; ?>

<?php echo " Esto es una prueba" ?>
```

Comentarios

PHP soporta los siguientes tipos de comentarios:

```
// Comentario

# Comentario

/*
    Comentario de
    varias líneas
*/
```

Tipos de Datos

El tipo de una variable normalmente no lo indica el programador; en su lugar, lo decide PHP en tiempo de ejecución dependiendo del contexto en el que se utilice esa variable. Si se quisiese obligar a que una variable se convierta a un tipo concreto, se podría forzar la variable o usar la función settype() para ello.

Enteros

Los enteros se puede especificar usando una de las siguientes sintaxis:

```
$a = 1234; # número decimal
$a = -123; # un número negativo
$a = 0123; # número octal (equivalente al 83 decimal)
$a = 0x12; # número hexadecimal (equivalente al 18 decimal)
```

Números en punto flotante

Los números en punto flotante ("double") se pueden especificar utilizando cualquiera de las siguientes sintaxis:

```
$a = 1.234; $a = 1.2e3;
```

Cadenas

Las cadenas de caracteres se pueden especificar usando uno de dos tipos de delimitadores.

Si la cadena está encerrada entre comillas dobles ("), las variables que estén dentro de la

cadena serán expandidas (sujetas a ciertas limitaciones de interpretación). Como en C y en Perl, el carácter de barra invertida (\ ) se puede usar para especificar caracteres especiales como:

\n Nueva línea
\r Retorno de carro
\t Tabulación horizontal
\\ Barra invertida
\$ Signo de pesos
\" Comillas dobles

Se puede proteger cualquier otro carácter, pero se producirá una advertencia en el nivel de depuración más alto.

La segunda forma de delimitar una cadena de caracteres usa el carácter de comilla simple ('). Cuando una cadena va encerrada entre comillas simples, los únicos caracteres de escape que serán comprendidos son \\ y \'. Esto es por convenio, así que se pueden tener comillas simples y barras invertidas en una cadena entre comillas simples. Las variables no se expandirán dentro de una cadena entre comillas simples.

Otra forma de delimitar cadenas es usando la sintaxis de documento incrustado ("<<<"). Se debe proporcionar un identificador después de <<<, después la cadena, y después el mismo identificador para cerrar el entrecomillado.

Ejemplos:

```
echo $str="Cadena con expansion de \"variables\" ";

echo "<br><br>";

echo $str='Cadena sin expansion de \'variables\' ';

echo "<br><br>";

echo <<<FDL
Ejemplo de cadena
Expandiendo múltiples líneas
usando sintaxis de documento incrustado.
FDL;
```

Arrays

Arrays unidimensionales.

PHP soporta tanto arrays escalares como asociativos. De hecho, no hay diferencias entre los dos. Se puede crear una array usando las funciones list() o array(), o se puede asignar el valor de cada elemento del array de manera explícita.

```
$a[0] = "abc";
$a[1] = "def";
$b["foo"] = 13;
```

También se puede crear un array simplemente añadiendo valores al array. Cuando se asigna un valor a una variable array usando corchetes vacíos, el valor se añadirá al final del array.

```
$a[] = "hola"; // $a[2] == "hola"
$a[] = "mundo"; // $a[3] == "mundo"
```

Arrays Multidimensionales

Los arrays multidimensionales son bastante simples actualmente. Para cada dimensión del array, se puede añadir otro valor [clave] al final:

```
$a[1] = $f; # ejemplos de una sola dimensión
$a["foo"] = $f;
$a[1][0] = $f; # bidimensional
$a["foo"][2] = $f; # (se pueden mezclar índices numéricos y asociativos)
$a[3]["bar"] = $f; # (se pueden mezclar índices numéricos y asociativos)
$a["foo"][4]["bar"][0] = $f; # tetradimensional
```

Objetos

Para inicializar un objeto, se usa la sentencia new para instanciar el objeto a una variable.

```
class clase
{
    function metodo ()
    {
        echo "Hola Mundo con PHP y POO.";
    }
}
```

```
$O = new clase;
$O->metodo();
```

Variables.

En PHP las variables se representan como un signo de pesos seguido por el nombre de la variable. El nombre de la variable es sensible a minúsculas y mayúsculas.

Los nombres de variables siguen las mismas reglas que otras etiquetas en PHP. Un nombre de variable valido tiene que empezar con una letra o un guion bajo, seguido de cualquier número de letras.

Nota: En nuestro caso, una letra es a-z, A-Z, y los caracteres ASCII del 127 al 255 (0x7f-0xff).

```
<?php
$var = "Bob";
$Var = "Joe";
echo "$var, $Var";      // saldria "Bob, Joe"

$4site = 'no funciona';     // varable no valida

$_4site = 'a ha';    // valid; starts with an underscore
$täyte = 'que cosa';    // valid; 'ä' is ASCII 228 (Extendido)
?>
```

Constantes.

El nombre de una constante sigue las mismas reglas que cualquier etiqueta en PHP. Un nombre de constante válido empieza con una letra o un caracter de subrayado, seguido por cualquier número de letras, números, o subrayados.

Se puede definir una constante usando la función define(). Una vez definida, no puede ser modificada ni eliminada .

Solo se puede definir como constantes valores escalares (boolean, integer, float y string).

Para obtener el valor de una constante solo es necesario especificar su nombre. A diferencia de las variables, no se tiene que especificar el prefijo $.

Estas son las diferencias entre constantes y variables:

Las constantes no son precedidas por un símbolo de pesos ($)

Las contantes solo pueden ser definidas usando la función define , nunca por simple asignación.

Las constantes pueden ser definidas y accedidas sin tener en cuenta las reglas de alcanze del ámbito.

Las constantes no pueden ser redefinidas o eliminadas despues de establecerse; y

Las constantes solo puede albergar valores escalares.

Ejemplo:

```
define("CONSTANTE", "Hola Mundo con constantes.");
echo CONSTANTE;
```

Operadores.

Operadores Aritméticos

$a + $b Adición Suma de $a y $b.
$a - $b Substracción Diferencia entre $a y $b.
$a * $b Multiplicación Producto de $a y $b.
$a / $b División Cociente de $a entre $b.
$a % $b Módulo Resto de $a dividido entre $b.

Operadores de Asignación

El operador básico de asignación es "=". A primera vista podrías pensar que es el operador de comparación "igual que". Pero no. Realmente significa que el operando de la izquierda toma el valor de la expresión a la derecha, (esto es, "toma el valor de").

El valor de una expresión de asignación es el propio valor asignado. Esto es, el valor de "$a = 3" es 3. Esto permite hacer cosas curiosas como

```
$a = ($b = 4) + 5; // ahora $a es igual a 9, y $b vale 4.
```

Además del operador básico de asignación, existen los "operadores combinados" para todas las operaciones aritméticas y de cadenas que sean binarias. Este operador combinado te permite, de una sola vez, usar una variable en una expresión y luego establecer el valor de esa variable al resultado de la expresión. Por ejemplo:

```
$a = 3;
```

$a += 5; // establece $a a 8, como si hubiésemos escrito: $a = $a + 5;
$b = "Hola ";
$b .= "Ahí!"; // establece $b a "Hola Ahí!", igual que si hiciésemos $b = $b . "Ahí!";

Operadores Bit a bit

Los operadores bit a bit te permiten activar o desactivar bits individuales de un entero.

$a & $b Y Se activan los bits que están activos tanto en $a como $b.
$a | $b O Se activan los bits que están activos en $a o que lo están en $b.
$a ^ $b Xor ("o exclusiva") Se activan los bits que están activos en $a o en $b pero no en ambos a la vez.
~ $a No Se activan los bits que no están activos en $a.
$a << $b Desplazamiento a la izquierda Desplaza los bits de $a, $b posiciones hacia la izquierda (por aritmética binaria, cada posición desplazada equivale a multiplicar por dos el valor de $a)
$a >> $b Desplazamiento a la derecha Desplaza los bits de $a, $b posiciones hacia la derecha (por aritmética binaria, cada posición desplazada equivale a dividir entre dos el valor de $a)

Operadores de Comparación

Los operadores de comparación, como su nombre indica, permiten comparar dos valores.

$a == $b Igualdad Cierto si $a es igual a $b.
$a === $b Identidad Cierto si $a es igual a $b y si son del mismo tipo (sólo PHP4)
$a != $b Desigualdad Cierto si $a no es igual a $b.
$a < $b Menor que Cierto si $a es estrictamente menor que $b.
$a > $b Mayor que Cierto si $a es estrictamente mayor que $b.
$a <= $b Menor o igual que Cierto si $a es menor o igual que $b.
$a >= $b Mayor o igual que Cierto si $a es mayor o igual que $b.

Otro operador condicional es el operador "?:" (o ternario), que funciona como en C y otros muchos lenguajes.

(expr1) ? (expr2) : (expr3);

La expresión toma el valor expr2 si expr1 se evalúa a cierto, y expr3 si expr1 se evalúa a falso.

Operador de ejecución

PHP soporta un operador de ejecución: la comilla invertida (``). PHP intentará ejecutar la instrucción contenida dentro de las comillas invertidas como si fuera un comando del shell; y su salida devuelta como el valor de esta expresión.

```
$output = `ls -al`;
echo "<pre>$output</pre>";
```

Operadores de Incremento/decremento

PHP soporta los operadores de pre decremento y post incremento al estilo de C.

++$a Preincremento Incrementa $a en uno y después devuelve $a.
$a++ Postincremento Devuelve $a y después incrementa $a en uno.
--$a Predecremento Decrementa $a en uno y después devuelve $a.
$a-- Postdecremento Devuelve $a y después decrementa $a en uno.

Operadores Lógicos

$a and $b Y Cierto si tanto $a como $b son ciertos.
$a or $b O Cierto si $a o $b son ciertos.
$a xor $b O exclusiva Cierto si $a es cierto o $b es cierto, pero no ambos a la vez.
! $a Negación Cierto si $a no es cierto.
$a && $b Y Cierto si tanto $a como $b son ciertos.
$a || $b O Cierto si $a o $b son ciertos.

Precedencia de Operadores

La siguiente lista muestra la precedencia de operadores, indicándose primero los de menor precedencia.

```
,
or
xor
and
print
= += -= *= /= .= %= &= |= ^= ~= <<= >>=
? :
||
&&
|
^
&
== != ===
< <= > >=
<< >>
+ - .
* / %
```

! ~ ++ -- (int) (double) (string) (array) (object) @
[
new

Operadores de Cadenas

Hay dos operadores de cadenas. El primero es el operador de concatenación (.), que devuelve el resultado de concatenar sus operandos izquierdo y derecho. El segundo es el operador de concatenación y asignación (.=).

```
$a = "Hola ";
$b = $a . "Mundo!";

$a = "Hola ";
$a .= "Mundo!";
```

Estructuras de Control.

if

```
if ($a > $b) {
    print "a es mayor que b";
 }
```

else

```
if ($a > $b) {
    print "a es mayor que b";
 }else{
    print "a NO es mayor que b";
}
```

elseif

```
if ($a > $b) {
    print "a es mayor que b";
 }elseif ($a < $b) {
    print "a es menor que b";
} else{
    print "a es igual que b";
}
```

while

```
$i = 1;
 while ($i <= 10) {
     echo $i++;
 }
```

do..while

```
$i = 1;
do {
     echo $i++;
 } while ($i <= 10);
```

for

```
for ($i = 1; $i <= 10; $i++) {
     print $i;
 }
```

foreach

PHP4 (PHP3 no) incluye una construcción foreach, tal como perl y algunos otros lenguajes. Esto simplemente da un modo fácil de iterar sobre arrays.

```
foreach( $arr as $value ) {
   echo "Valor: $value<br>\n";
}
```

break

Escapa de la estructuras de control iterante (bucle) actuales for, while, o switch.

```
$val=1;
while ($val<20) {
    $val++;
    if ($val == 15) {
        break;
    }
    echo "$val<br>\n";
}
```

continue

Se usa dentro de la estructura del bucle para saltar el resto de la iteración actual del bucle y continuar la ejecución al comienzo de la siguiente iteración.

```
while ($val<20) {
   $val++;
   if ($val == 15) {
      continue;
   }
   echo "$val<br>\n";
}
```

switch

```
switch ($i) {
    case 0:
       print "i es igual a 0";
       break;
    case 1:
       print "i es igual a 1";
       break;
    case 2:
       print "i es igual a 2";
       break;
 }
```

Funciones.

Funciones definidas por el usuario

```
function Saludo () {
        echo “Hola Mundo con funciones de Usuario de PHP”;
}
Saludo();
```

Parámetros de las funciones

La información puede suministrarse a las funciones mediante la lista de parámetros, una lista de variables y/o constantes separadas por comas.

```
function Suma($a, $b) {
        $r = $a + $b;
        echo $r;
}
```

Por defecto, los parámetros de una función se pasan por valor. Si se requieren parámetros pro referencia, puedes anteponer un ampersand (&) al nombre del parámetro en la definición de la función.

```
function Suma(&$a, &$b) {
        $r = $a + $b;
        echo $r;
}
```

Una función puede definir valores por defecto para los parámetros escalares estilo C++.

```
function cafe ($taza = "capuchino") {
   return "Hacer una taza de $taza.\n";
}
echo cafe ();
echo cafe ("de cuca");
```

Devolver valores

Los valores se retornan usando la instrucción opcional return. Puede devolverse cualquier tipo de valor, incluyendo listas y objetos.

```
function cuadro ($num) {
   return $num * $num;
}
echo cuadro (4);
```

No puedes devolver múltiples valores desde una función, pero un efecto similar se puede conseguir devolviendo una lista.

Clases y Objetos.

Las clases en PHP se definen de la siguiente manera.

```
class clase_iva {
    //Propiedades de la clase
    var $cantidad;
    var $porcentaje;
    var $iva;
    //Constructor de la clase
    function clase_iva() {
        $this->porcentaje=0.15;
    }
    //Metodos de la clase
    function calcular_iva() {
        $this->iva=$this->porcentaje * $this->cantidad;
    }
}

class clase_isr extends clase_iva {
    var $isr;
    function calcular_isr() {
        $this->isr=$this->cantidad*0.1;
    }

}

$OF = new clase_isr;

$OF->cantidad=100;

$OF->calcular_iva();

$OF->calcular_isr();

echo “El IVA es: ”;

echo $OF->iva;

echo “<br>El ISR ES: ”;

echo $OF->isr;
```

Funciones de Cadenas.

echo

echo (string arg1 [, string argn...])

Da salida a todos sus parámetros.

echo() no es realmente una función (es una sentencia del lenguaje) de modo que no se requiere el uso de los paréntesis.

```
echo "Hola Mundo con PHP";

echo "Esto se extiende
por varias líneas. Los saltos de línea
también se envían";

echo "Esto se extiende\npor varias líneas. Los saltos de línea\ntambién se envían";
```

nl2br

nl2br -- Convierte nuevas líneas a saltos de línea HTML

string nl2br (string cadena)

Devuelve la cadena con '<BR>' insertados antes de cada nueva línea.

```
echo nl2br("Hola Mundo\ncon PHP\nFunciones");
```

strtolower y strtoupper

strtolower -- Pasa a minúsculas una cadena

string strtolower (string cad)

strtolower devuelve la cadena con todas sus letras en minúsculas.

strtoupper -- Pasa a mayúsculas una cadena

string strtoupper (string cadena)

strtoupper devuelve la cadena con todas sus letras en mayúsculas.

```
$cadena="HoLa mUnDo cOn pHp ";

echo strtolower($cadena).strtoupper($cadena);
```

ucfirst

ucfirst -- Pasar a mayúsculas el primer caracter de una cadena

string ucfirst (string cad)

Pone en mayúsculas el primer carácter de cad si es alfabético.

```
$texto = 'susanita tiene un ratón, un ratón chiquitito.';
echo $texto = ucfirst ($texto);
```

trim

trim -- Elimina espacios del principio y final de una cadena

string trim (string cad)

Esta función elimina los espacios en blanco del comienzo y del final de una cadena y devuelve el resultado. Los caracteres de espacio que elimina realmente son: "\n", "\r", "\t", "\v", "\0", y el espacio en sí.

strlen y str_replace

strlen -- Obtiene la longitud de la cadena

int strlen (string cad)

str_replace -- Sustitiye una cadena por otra dentro de una cadena

string str_replace (string busqueda, string remplazo, string cadena)

```
$cadena="Hola mundo Web con PHP";

$cadena=str_replace("mundo Web", "mundo del CONSOL 2004", $cadena);

echo $cadena." tiene ".strlen($cadena)." caracteres";
```

strrev

strrev -- Invierte una cadena

string strrev (string cadena)

explode e implode

explode -- Divide una cadena por otra

array explode (string separador, string cadena [, int limite])

Devuelve una matriz de cadenas, cada una de las cuales es una subcadena de cadena formada mediante su división en las fronteras marcadas por la cadena separador. Si se especifica limite, la matriz devuelta contendrá un máximo de limite elementos con el último conteniendo el resto de la cadena.

```
$pizza = "trozo1 trozo2 trozo3 trozo4 trozo5 trozo6";
$trozos = explode (" ", $pizza);
```

implode -- Unir elementos de una matriz mediante una cadena

string implode (string cola, array piezas)

Devuelve una cadena que contiene una representación de todos los elementos de la matriz en el mismo orden, pero con la cadena cola en medio de los mismos.

```
$pizza = implode (" ", $trozos);
```

Diversidad de Funciones de Cadenas
AddCSlashes -- Marca una cadena con barras al estilo del C
AddSlashes -- Marca una cadena con barras
bin2hex -- Convierte datos binarios en su representación hexadecimal
chop -- Elimina espacios sobrantes al final
chr -- Devuelve un caracter específico
chunk_split -- Divide una cadena en trozos más pequeños
convert_cyr_string -- Convierte de un juego de caracteres Cirílico a otro
count_chars -- Devuelve información sobre los caracteres usados en una cadena
crc32 -- Calcula el polinomio crc32 de una cadena
crypt -- Encripta una cadena mediante DES
echo -- Da salida a una o más cadenas
explode -- Divide una cadena por otra
fprintf -- Escribir una cadena con formato a una secuencia

get_html_translation_table -- Devuelve la tabla de traducción utilizada por htmlspecialchars() y htmlentities()
hebrev -- Convierte Hebreo lógico a texto visual
hebrevc -- Convierte Hebreo lógico a texto visual con conversión de saltos de línea
html_entity_decode -- Convertir todas las entidades HTML a sus caracteres correspondientes
htmlentities -- Convierte todos los caracteres aplicables a entidades HTML
htmlspecialchars -- Convierte caracteres especiales a entidades HTML
implode -- Unir elementos de una matriz mediante una cadena
join -- Une elementos de una tabla mediante una cadena
levenshtein -- Calcula la distancia Levenshtein entre dos cadenas
localeconv -- Obtener información sobre el formato numérico
ltrim -- Elimina el espacio en blanco del principio de una cadena
md5_file -- Calcula el resumen criptográfico md5 de un nombre de archivo dado
md5 -- Calcula el hash md5 de una cadena
metaphone -- Calcula la clave "metáfona" de una cadena
money_format -- Da formato a un número como una cadena de moneda
nl_langinfo -- Consultar información sobre el lenguaje y la localidad
nl2br -- Convierte nuevas líneas a saltos de línea HTML
number_format -- Dar formato a un número con los miles agrupados
ord -- Devuelve el valor ASCII de un caracter
parse_str -- Divide la cadena en variables
print -- Emite una cadena
printf -- Emite una cadena con formato
quoted_printable_decode -- Convierte una cadena con marcación imprimible a una cadena de 8 bits
quotemeta -- Quote meta characters
rtrim -- Elimina espacios en blanco al final de la cadena.
setlocale -- Fija la información de localidad
sha1_file -- Calcular el resumen criptográfico sha1 de un archivo
sha1 -- Calcular el resumen criptográfico sha1 de una cadena
similar_text -- Calcula la similitud entre dos cadenas
soundex -- Calcula la clave soundex de una cadena
sprintf -- Devuelve una cadena con formato
sscanf -- Trocea la entrada desde una cadena según un formato dado
str_ireplace -- Versión insensible a mayúsculas y minúsculas de str_replace().
str_pad -- Rellena una cadena con otra hasta una longitud dada
str_repeat -- Repite una cadena
str_replace -- Sustitiye todas las apariciones de la aguja en el pajar por la cadena
str_rot13 -- Realizar la transformación rot13 sobre una cadena
str_shuffle -- Reordena aleatoriamente una cadena
str_split -- Convertir una cadena en una matriz
str_word_count -- Devolver información sobre las palabras usadas en una cadena
strcasecmp -- Comparación de cadenas insensible a mayúsculas y minúsculas y segura en

modo binario
strchr -- Encuentra la primera aparición de un caracter
strcmp -- Comparación de cadenas con seguridad binaria
strcoll -- Comparación de cadenas basada en la localidad
strcspn -- Encuentra la longitud del elemento inicial que no coincide con la máscara
strip_tags -- Elimina marcas HTML y PHP de una cadena
stripcslashes -- Desmarca la cadena marcada con addcslashes()
stripos -- Encontrar la posición de la primera ocurrencia de una cadena, insensible a mayúsculas y minúsculas
stripslashes -- Desmarca la cadena marcada con addslashes()
stristr -- strstr() sin tener en cuenta mayúsculas o minúsculas
strlen -- Obtiene la longitud de la cadena
strnatcasecmp -- Comparación de cadenas insensible a mayúsculas y minúsculas usando un algoritmo de "orden natural"
strnatcmp -- Compara cadenas usando un algoritmo de "orden natural"
strncasecmp -- Comparación de los primeros n caracteres de cadenas, segura con material binario e insensible a mayúsculas y minúsculas
strncmp -- Comparación de los n primeros caracteres de cadenas, con seguridad binaria
strpos -- Encuentra la posición de la primera aparición de una cadena
strrchr -- Encuentra la última aparición de un caracter en una cadena
strrev -- Invierte una cadena
strripos -- Encontrar la posición de la última ocurrencia de una cadena en otra, insensible a mayúsculas y minúsculas
strrpos -- Encuentra la posición de la última aparición de un caracter en una cadena
strspn -- Encuentra la longitud del segmento inicial que coincide con la máscara
strstr -- Encuentra la primera aparición de una cadena
strtok -- Divide una cadena en elementos
strtolower -- Pasa a minúsculas una cadena
strtoupper -- Pasa a mayúsculas una cadena
strtr -- Traduce ciertos caracteres
substr_compare -- Comparación de 2 cadenas, segura con material binario, opcionalmente insensible a mayúsculas y minúsculas, a partir de un desplazamiento, y hasta un número límite de caracteres
substr_count -- Cuenta el número de apariciones de la subcadena
substr_replace -- Sustituye texto en una parte de una cadena
substr -- Devuelve parte de una cadena
trim -- Elimina espacios del principio y final de una cadena
ucfirst -- Pasar a mayúsculas el primer caracter de una cadena
ucwords -- Pone en mayúsculas el primer caracter de cada palabra de una cadena
vprintf -- Imprimir una cadena con formato
vsprintf -- Devuelve una cadena con formato
wordwrap -- Corta una cadena en un número dado de caracteres usando un caracter de ruptura de cadenas.

Funciones de Variables.

empty, isset y unset

empty -- Determina si una variable está definida

int empty (mixed var)

Devuelve FALSE si var está definida y tiene un valor no-vacío o distinto de cero; en otro caso devuelve TRUE.

isset -- Determina si una varible está definida

int isset (mixed var)

Devuelve TRUE si var existe; y FALSE en otro caso.

unset -- Destruye una variable dada

int unset (mixed var)

unset() destruye la variable especificada y devuelve TRUE.

```
$var=1;
do{
    $var++;
    if(empty($rr))
      $rr=0;
    if(isset($rr))
      $rr+=2;

    echo $rr;

    if($rr>4)
      unset($rr);

}while($var<=5);
```

print_r

print_r -- Imprime información legible para humanos sobre una variable

bool print_r (mixed expresion [, bool devolver])

Nota: El parámetro devolver fue agregado en PHP 4.3.0

print_r() despliega información sobre una variable en una forma que es apta para su lectura por humanos. Si se le entrega una variable tipo string, integer o float, el valor mismo será impreso. Si se le entrega un array, los valores serán presentados en un formato que muestra las claves y los elementos. Una notación parecida es usada para variables tipo object.

Recuerde que print_r() desplazará el apuntador de la matriz al final. Use reset() para llevarlo de vuelta al comienzo.

```
$l[0]=1;$l[1]=2;$l[2]=3;$l["yo"]="hector";
echo "<pre>";
print_r($l);
echo "</pre>";
```

Diversidad de funciones de variables

doubleval -- Obtiene el valor double (decimal) de una variable.
empty -- Determina si una variable está definida
floatval -- Obtener el valor flotante de una variable
get_defined_vars -- Devuelve una matriz con todas la variables definidas
get_resource_type -- Devuelve el tipo de recurso
gettype -- Obtiene el tipo de una variable.
import_request_variables -- Importar variables GET/POST/Cookie en el contexto global
intval -- Obtiene el valor entero de una variable.
is_array -- Averigua si una variable es un array.
is_bool -- Encuentra si una variable es de tipo booleana
is_callable -- Verifica que los contenidos de una variable puedan ser llamados como una función
is_double -- Averigua si una variable es un valor double (número decimal).
is_float -- Averigua si una variable es un flotante.
is_int -- Averigua si una variable es un valor entero.
is_integer -- Averigua si una variable es un valor entero.
is_long -- Averigua si una varible es un valor entero.
is_null -- Encuentra si una variable es NULL
is_numeric -- Encuentra si una variable es un número o una cadena numérica
is_object -- Averigua si una varible es un objeto.
is_real -- Averigua si una varible es un número real.
is_resource -- Encuentra si una variable es un recurso
is_scalar -- Encuentra si una variable es un escalar
is_string -- Averigua si una varible es una cadena de caracteres (string).
isset -- Determina si una varible está definida
print_r -- Imprime información legible para humanos sobre una variable

serialize -- Genera una representación apta para almacenamiento de un valor
settype -- Establece el tipo de una variable.
strval -- Obtiene una cadena de caracteres a partir de una variable.
unserialize -- Crea un valor PHP a partir de una representación almacenada
unset -- Destruye una variable dada
var_dump -- Vuelca información sobre una variable
var_export -- Imprime o devuelve la representación de cadena de una variable

Funciones de Matrices.

array

array -- Crear una matriz

array array (mixed ...)

Devuelve una matriz con los parámetros que se le pasan. A dichos parámetros se les puede dar un índice usando el operador =>.

Nota: array() es una construcción del lenguaje que se utiliza para representar matrices literales, no una función regular.

```
$utiles=array ("cuaderno",4=>"libro","lapiz");
```

El siguiente ejemplo demuestra cómo crear una matriz bidimensional, cómo especificar claves para matrices asociativas, y cómo especificar índices no consecutivos en matrices normales.

```
$frutas = array (
    "frutas"  => array("a"=>"naranja", "b"=>"plátano", "c"=>"manzana"),
    "números" => array(1, 2, 3, 4, 5, 6),
    "hoyos"   => array("primero", 5 => "segundo", "tercero")
);
```

count, array_keys y array_values

count -- Cuenta los elementos de una variable

int count (mixed var)

Devuelve el número de elementos en var, que típicamente es una matriz (porque cualquier otra cosa tendría sólo un elemento).

Devuele 1 si la variable no es una matriz.

Devuelve 0 si la variable no tiene valor.

array_keys -- Devuelve todas las claves de una matriz

array array_keys (array entrada [, mixed val_a_buscar])

array_keys() devuelve las claves, numéricas y de cadena, de la matriz entrada.

Si se especifica el parámetro opcional val_a_buscar, sólo se devuelven las claves para dicho valor. De otro modo, se devuelven todas las claves de la entrada.

array_values -- Devuelve todos los valores de una matriz

array array_values (array entrada)

array_values() devuelve todos los valores de la matriz entrada.

```
$arreglo["Pan"]="Concha";
$arreglo["Color"]="Rojo";
$arreglo[16]="Hola Mundo con PHP";
$arreglo[1]=3.1416326420;

$llaves=array_keys($arreglo);
$valores=array_values($arreglo);
$elementos=count($llaves);
for($i=0;$i<$elementos;$i++) {
    echo "<br>El arreglo[ " . $llaves[$i] . " ]= " . $valores[$i];
}
```

each y list

each -- Devuelve el siguiente par clave/valor de una matriz

array each (array matriz)

Devuelve el par clave/valor actual para la matriz y avanza el cursor de la misma. Esta pareja se devuele en una matriz de 4 elementos, con las claves 0, 1, key, y value. Los elementos 0 y key contienen el nombre de clave del elemento de la matriz, y 1 y value contienen los datos.

Si el puntero interno para la matriz apunta pasado el final del contenido de la matriz, each() devuelve FALSE.

list -- Asigna variables como si fueran una matriz

void list (mixed ...)

Como array(), esta no es realmente una función, sino una construcción del lenguaje. list() se usa para asignar una lista de variables en una sola operación.

```
$arreglo=array("Pan"=>"Concha","Color"=>"Rojo",16=>"Hola Mundo con PHP",1=>3.1416326420);

while(list($llave,$valor)=each($arreglo)) {
    echo "<br>El arreglo [ ". $llave ." ]= ".$valor;
}
```

Diversidad de Funciones de Matrices

array_change_key_case -- Devuelve una matriz con todas las claves de las cadenas convertidas a mayúsculas o minúsculas
array_chunk -- Divide una matriz en segmentos
array_combine -- Crea una nueva matriz, usando una matriz para las claves y otra para sus valores
array_count_values -- Cuenta todos los valores de una matriz
array_diff_assoc -- Comprueba las diferencias entre matrices teniendo en cuenta los índices
array_diff_uassoc -- Computa la diferencia entre matrices con un chequeo adicional de índices, el cual es realizado por una llamada de retorno entregada por el usuario.
array_diff -- Comprueba las diferencias entre matrices
array_fill -- Llena una matriz con valores
array_filter -- Filtra elementos de una matriz mediante una función "callback"
array_flip -- Intercambia los valores de una matriz
array_intersect_assoc -- Computes the intersection of arrays with additional index check
array_intersect -- Computes the intersection of arrays
array_key_exists -- Comprueba si el índice o clave dada existe en la matriz
array_keys -- Devuelve todas las claves de una matriz
array_map -- Aplica la llamada de retorno especificada a los elementos de las matrices dadas
array_merge_recursive -- Une dos o más matrices recursivamente
array_merge -- Combina dos o más matrices
array_multisort -- Ordena múltiples matrices, o matrices multi-dimensionales
array_pad -- Rellena una matriz con un valor hasta el tamaño especificado
array_pop -- Extrae el último elemento de la matriz
array_push -- Inserta uno o más elementos al final de la matriz
array_rand -- Selecciona una o más entradas aleatorias de una matriz
array_reduce -- Reduce iterativamente una matriz a un solo valor usando una función llamada de retorno

array_reverse -- Devuelve una matriz con los elementos en orden inverso
array_search -- Busca un valor determinado en una matriz y devuelve la clave correspondiente en caso de éxito
array_shift -- Extrae un elemento del comienzo de la matriz
array_slice -- Extrae una porción de la matriz
array_splice -- Suprime una porción de la matriz y la sustituye por otra cosa
array_sum -- Calcula la suma de los valores en una matriz
array_udiff_assoc -- Computa la diferencia entre matrices con un chequeo de índices adicional. Los datos son comparados usando una llamada de retorno.
array_udiff_uassoc -- Computa la diferencia entre matrices con un chequeo de índices adicional. Los datos son comparados usando una llamada de retorno. El chequeo de índice es realizado por una llamada de retorno también.
array_udiff -- Computa la diferencia entre matrices, usando una llamada de retorno para la comparación de datos.
array_unique -- Remueve valores duplicados de una matriz
array_unshift -- Introduce uno o más elementos al principio de la matriz
array_values -- Devuelve todos los valores de una matriz
array_walk -- Aplica una función del usuario a cada elemento de una matriz.
array -- Crear una matriz
arsort -- Ordena una matriz en orden inverso y mantiene la asociación de índices
asort -- Ordena una matriz y mantiene la asociación de índices
compact -- Crea una matriz que contiene variables y sus valores
count -- Cuenta los elementos de una variable
current -- Devuelve el elemento actual de una matriz
each -- Devuelve el siguiente par clave/valor de una matriz
end -- Mueve el puntero interno de una tabla al último elemento
extract -- Importa variables a la tabla de símbolos desde una matriz
in_array -- Devuelve TRUE si un valor está en una matriz
key -- Obtiene una clave de una matriz asociativa
krsort -- Ordena una matriz por clave en orden inverso
ksort -- Ordena una matriz por clave
list -- Asigna variables como si fueran una matriz
natcasesort -- Ordena una matriz usando un algoritmo de "orden natural" sin distinguir mayúsculas de minúsculas
natsort -- Ordena una matriz usando un algoritmo de "orden natural"
next -- Avanza el puntero interno de una matriz
pos -- Obtiene el elemento actual de una matriz
prev -- Rebobina el puntero interno de una matriz
range -- Crea una matriz que contiene un rango de elementos
reset -- Fija el puntero interno de una matriz a su primer elemento
rsort -- Ordena una matriz en orden inverso
shuffle -- Mezcla una matriz
sizeof -- Obtiene el número de elementos de una matriz

sort -- Ordena una matriz
uasort -- Ordena una matriz mediante una función de comparación definida por el usuario y mantiene la asociación de índices
uksort -- Ordena una matriz por claves mediante una función definida por el usuario
usort -- Ordena una matriz por valores mediante una funcion definida por el usuario

Funciones de Fecha y Hora.

time y getdate

time -- devuelve el timestamp UNIX actual

int time (void)

Devuelve la hora actual como número de segundos transcurridos desde las 00:00:00 del 1 de enero de 1970 GMT (Unix Epoch).

getdate -- obtiene información de fecha y hora

array getdate (int timestamp)

Devuelve un array asociativo que contiene la información de fecha del valor timestamp como los siguientes elementos:

"seconds" - segundos

"minutes" - minutos

"hours" - horas

"mday" - día del mes

"wday" - día de la semana, en número

"mon" - mes, en número

"year" - año, en número

"yday" - día del año, en número; por ejemplo, "299"

"weekday" - día de la semana, en texto, completo; por ejemplo, "Friday"

"month" - mes, en texto, completo; por ejemplo, "January"

```
echo "<pre>";
print_r(getdate(time()));
echo "</pre>";
```

Diversidad de Funciones de Fecha y Hora

checkdate -- valida una fecha u hora
date -- da formato a la fecha/hora local
getdate -- obtiene información de fecha y hora
gettimeofday -- obtiene la hora actual
gmdate -- da formato a una fecha/hora GMT/CUT
gmmktime -- obtiene el valor timestamp UNIX de una fecha GMT
gmstrftime -- da formato a una fecha/hora GMT/CUT según las convenciones locales
localtime -- Obtener la hora local
microtime -- devuelve el valor timestamp UNIX actual con microsegundos
mktime -- obtiene el timestamp UNIX de una fecha
strftime -- da formato a la hora o fecha local de acuerdo con las convenciones locales
strtotime -- Procesar cualquier descripción textual de fecha/hora en Inglés convirtiéndola en una timestamp de UNIX.
time -- devuelve el timestamp UNIX actual

Funciones de Correo.

mail

mail -- Envía correo

bool mail (string para, string sobre, string mensaje [, string cabeceras_adicionales])

mail() envía automáticamente el mensaje especificado en mensage al destinatario especificado en para. Para especificar múltiples destinatarios se puede hacer incluyendo una coma entre las direcciones en para. Ejemplo.

```
mail("hector@interlegit.com.mx", "Sobre este tema", "Hola 1\nMundo 2\ncon 3\nPHP");
```

Si se añadiera una cadena como cuarto argumento, esta cadena sería enviada al final de la cabecera. Esto se usa normalmente para enviar cabeceras extra en los mensajes. Si se desea enviar varias cabeceras extra el mecanismo será el mismo separándolas una linea. Ejemplo.

```
$destino="hector@interlegit.com.mx";
```

```
$tema="Segundo ejemplo";
$mensaje="Este es el mensaje, Hola Mundo con PHP";
$cabeceras="From: webmaster@interlegit.com.mx\nReply-To:webmaster@interlegit.com.mx\nX-Mailer: PHP/".phpversion();
mail($destino,$tema,$mensaje,$cabeceras);
```

Diversidad de Funciones de Correo

ezmlm_hash -- Calcular el valor hash que necesita EZMLM
mail -- Envía correo

Funciones de Ejecución de Programas.

shell_exec

shell_exec -- Ejecutar un comando mediante el intérprete de comandos y devolver la salida completa como una cadena

string shell_exec (string cmd)

exec

exec -- Ejecuta un programa externo

string exec (string command [, string array [, int return_var]])

exec() ejecuta la orden indicada en command, sin embargo no produce ninguna salida. Simplemente devuelve la última línea de la salida resultado de la orden.

Si el parámetro array existe, entonces el array especificado se rellenará con cada una de las líneas de la salida producida por la orden.

Si el parámetro return_var existe a la vez que el parámetro array, entonces el valor de retorno de la orden ejecutada se guardará en dicha variable. Ejemplos.

```
echo shell_exec("ls");

$coman= exec ("ls -laiF",$salida,$pp);
$ciclos=count($salida);
for($i=0;$i<$ciclos;$i++){
        echo "<br>salida[$i]=".$salida[$i];
}
```

Diversidad de Funciones de Ejecucion de Programas

escapeshellarg -- escapar una cadena a ser usada como argumento del intérprete de comandos
escapeshellcmd -- enmascara los metacaracteres del intérprete de ordenes
exec -- Ejecuta un programa externo
passthru -- Ejecuta un programa externo y muestra su salida literal
proc_close -- Cierra un proceso abierto por proc_open() y devuelve el código de salida del proceso.
proc_get_status -- Obtiene información sobre un proceso abierto por proc_open()
proc_nice -- Modificar la prioridad del proceso actual
proc_open -- Ejecutar un comando y abrir apuntadores de archivo para entrada/salida
proc_terminate -- mata un proceso abierto por proc_open
shell_exec -- Ejecutar un comando mediante el intérprete de comandos y devolver la salida completa como una cadena
system -- Ejecuta un programa externo y muestra su salida

Descripción y demostración Facileal.

Facileal, es una aplicación OpenSource, que te permite tener un sitio en cuestión de minutos de una manera muy fácil y muy sencilla, además su administración es muy amigable, y por si fuera poco Facileal es muy fácil de navegar.

Facileal esta desarrollado para aquellas personas que no tienen conocimientos de paginas Web y desean tener su propio sitio en poco tiempo, ya que no es necesario tener conocimientos de PHP, ni de Bases de Datos, lo único que debes hacer es cargarlo en tu servidor de Hospedaje y listo, podrás administrarlo.

Facileal, también es multiplataforma lo que quiere decir que lo puedes utilizar en servidores Windows, Linux o Unix.

Facileal es una pequeña aplicación realizada con PHP que utiliza el modelo de programación Orientado a Objetos, así que para aquellas personas que gustan de la programación Web, Facileal es una aplicación fácil de entender y por lo tanto fácil de modificar, para que puedas adaptarlo a tu gusto.

Puedes visitar la página de Facileal en:

http://facileal.interlegit.com.mx

Aplicable a la Educación y Aplicable al Comercio.

Aplicable al Desarrollo de Sitios en General.

El Senado de la República, el 24 de Abril del 2003, estaba funcionando con la siguiente página web, que es rica en imágenes, animaciones y contenido. Facileal, puede adaptarse en pocos minutos a diseños similares a ese, con gran contenido informativo, aunque para implementar imágenes o flash, hará falta conocer un poco de HTML Básico.